

**Project Report
PCA-IRT-6**

**Morphing Scenarios for the GMTI Portion
of the PCA Integrated Radar Tracker**

W.G. Coate
J.M. Lebak

20 February 2004
Issued: 21 December 2004

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS



Prepared for the Defense Advanced Research Projects Agency
under Air Force Contract F19628-00-C-0002.

Approved for public release; distribution is unlimited.


This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. This work was sponsored by the Defense Advanced Research Projects Agency, IPTO, under Air Force Contract F19628-00-C-0002.

This report may be reproduced to satisfy needs of U.S. Government agencies.

The ESC Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER


Gary Tutungian
Administrative Contracting Officer
Plans and Programs Directorate
Contracted Support Management

Non-Lincoln Recipients

PLEASE DO NOT RETURN

Permission is given to destroy this document
when it is no longer needed.

Massachusetts Institute of Technology
Lincoln Laboratory

Morphing Scenarios for the GMTI Portion of the
PCA Integrated Radar Tracker

W.G. Coate
J.M. Lebak
Group 102

Project Report PCA-IRT-6

20 February 2004
Issued: 21 December 2004

Approved for public release; distribution is unlimited.

Lexington

Massachusetts

ABSTRACT

The key to the success of the DARPA-sponsored Polymorphous Computing Architectures (PCA) program is the proper development of the morphing concept. One of MIT Lincoln Laboratory's contributions towards this effort is the Integrated Radar-Tracker (IRT) application. The initial IRT consists of a Ground Moving Target Indicator (GMTI) radar and a kinematic tracker (KT). As GMTI has received more community interest, it is used in this first document as a background for identification and discussion of promising computational issues which may be addressed by morphing. This document examines three areas from GMTI: data dependent load balancing; changing from streaming to threaded computation; and changing parameter sets.

TABLE OF CONTENTS

	Page
Abstract	ii
List of Illustrations	iv
List of Tables	iv
1. INTRODUCTION	1
1.1 Ground Moving Target Indicator Radar Overview	1
1.2 Mapping Considerations	4
1.3 Morphing Motivation	5
2. MORPH CHANGE A: TARGET NUMBER AND DISTRIBUTION CHANGE	7
3. MORPH CHANGE B: STREAM- TO THREAD-BASED COMPUTATION	10
4. MORPH CHANGE C: PARAMETER SET CHANGE	12
5. CONCLUSION	15
References	16

LIST OF ILLUSTRATIONS

Figure No.		Page
1	GMTI processing chain.	2
2	GMTI data cube illustration.	4
3	Target number and distribution morph change sample view. Bold lines indicate separate data cube areas. Thin lines indicate data slices for CFAR. Dots indicate targets.	7
4	Time-shared resource mapping for STAP, Detection, and Estimation.	11
5	Flop comparison between Wilderness and City scenarios.	13

LIST OF TABLES

Table No.		Page
1	GMTI Parameter Descriptions	3
2	GMTI Data Sizes	3
3	Target Distribution Example	8
4	Example System Descriptions for Load Balancing	8
5	Success Metrics for Example Systems	9
6	Stream to Threat Example Computational Balance	10
7	GMTI Morph State Parameters	12
8	GMTI Morph State Requirements	12

1. INTRODUCTION

The key to the success of the DARPA-sponsored Polymorphous Computing Architectures (PCA) program is the proper development of the morphing concept. One of MIT Lincoln Laboratory's contributions towards this effort is the Integrated Radar-Tracker (IRT) application. The initial IRT consists of a Ground Moving Target Indicator (GMTI) radar and a kinematic tracker (KT). As GMTI has received more community interest, it is used in this first document as a background for identification and discussion of promising computational issues which may be addressed by morphing. This document examines three areas from GMTI: data dependent load balancing; changing from streaming to threaded computation; and changing parameter sets.

1.1 GROUND MOVING TARGET INDICATOR RADAR OVERVIEW

The basic deliverable of the IRT application consists of two distinct pieces, a Ground Moving Target Indicator radar and a kinematic tracker. Either piece may be run separately or combined by feeding GMTI output to KT as input. Complete descriptions of GMTI and KT may be found in [3] and [2], respectively. A very high level overview of GMTI will be offered in this section.

1.1.1 GMTI Processing Stream

The GMTI implementation used for the IRT is composed of the seven stages shown in Figure 1 below. Time Delay & Equalization and Pulse Compression are FIR filters. Adaptive Beamforming and Space-Time Adaptive Processing (STAP) consist of QR/LQ, back/forward substitutions, and matrix multiplication. Doppler Filtering is essentially a Fast Fourier Transform (FFT). Detection consists of a Constant False-Alarm Rate (CFAR) thresholding operation and three-dimensional grouping (removing duplicate detections by only considering local maxima). Estimation incorporates spline interpolation and maximum likelihood estimation.

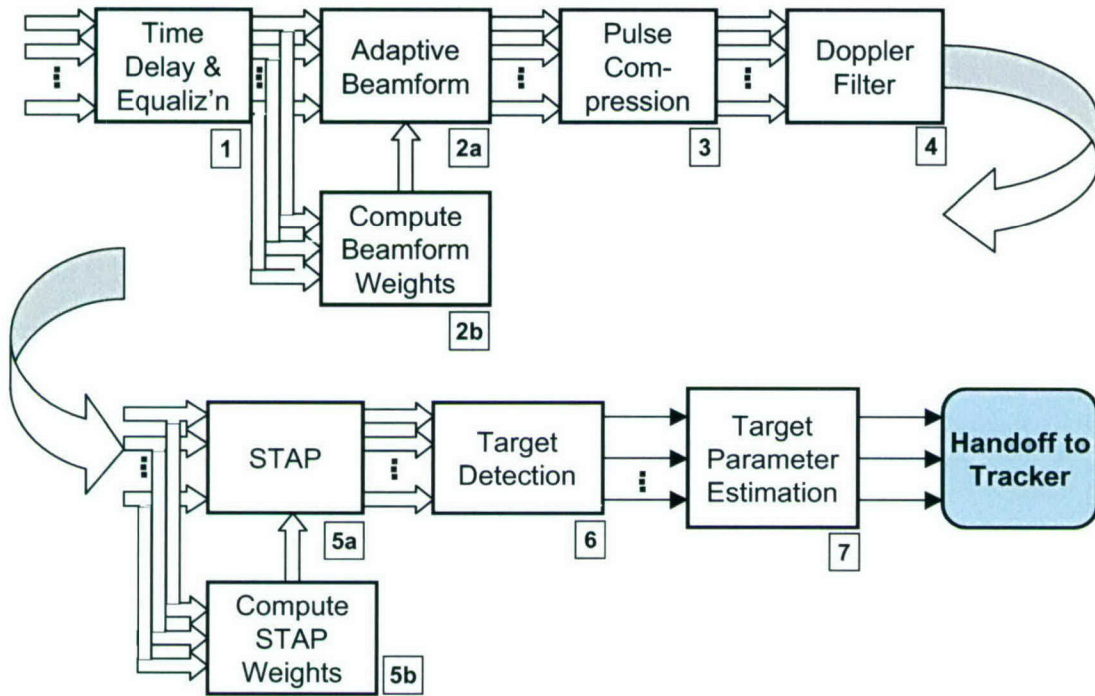


Figure 1. GMTI processing chain.

1.1.2 GMTI Data Flow

The data cube size is changed by each stage. In Table 1, we describe shortened names for the data size parameters; these names are used in the data flow description in Table 2. The axes of the data cube are illustrated by Figure 2. The target list size is obviously dependent upon the number of targets detected, so its size cannot be known a priori.

TABLE 1
GMTI Parameter Descriptions

Parameter Name	Parameter Description
Nrg	Number of range gates
Npc	Number of taps used in Pulse Compression filter
Nch	Number of channels (from antenna)
Nbm	Number of beams (from Adaptive Beamform)
Ncnb	Number of clutter-nulled beams (from STAP)
Npri	Number of pulses sent in one Pulse Repetition Interval (PRI)
Ndop	Number of Doppler bins
Nstag	Number of Doppler staggers calculated
Ntgts	Number of targets detected
TgtRptSize	Size of one target report

TABLE 2
GMTI Data Sizes

Stage	Input Data Size	Output Data Size
1 Time Delay & Equalization	$Nrg \times Nch \times Npri$	$Nrg \times Nch \times Npri$
2 Adaptive Beamform	$Nrg \times Nch \times Npri$	$Nrg \times Nbm \times Npri$
3 Pulse Compression	$Nrg \times Nbm \times Npri$	$(Nrg+Npc-1) \times Nbm \times Npri$
4 Doppler Filtering	$(Nrg+Npc-1) \times Nbm \times Npri$	$Nstag \times (Nrg+Npc-1) \times Nbm \times Ndop$
5 STAP	$Nstag \times (Nrg+Npc-1) \times Nbm \times Ndop$	$(Nrg+Npc-1) \times Ncnb \times Ndop$
6 Detection	$(Nrg+Npc-1) \times Ncnb \times Ndop$	$Ntgts \times TgtRptSize$
7 Estimation	$Ntgts \times TgtRptSize$, subset of STAP output	$Ntgts \times TgtRptSize$

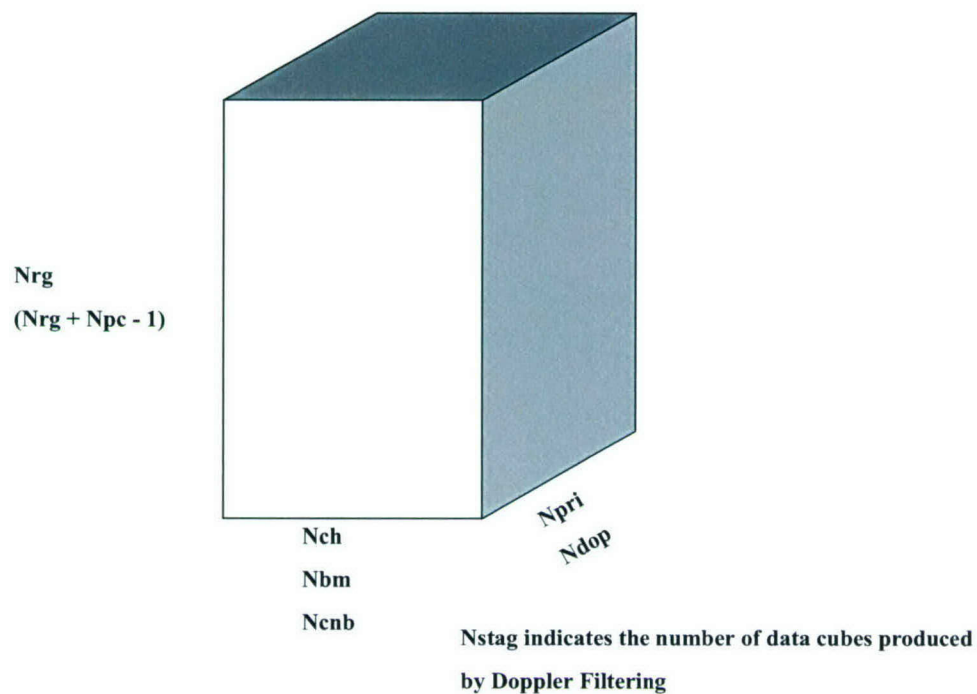


Figure 2. GMTI data cube illustration.

1.2 MAPPING CONSIDERATIONS

A real-time system is given a latency and a throughput requirement. Latency indicates the acceptable time lag between the data input and the result output. It may be specified as a number of coherent processing intervals (CPIs)¹, or sometimes by a certain absolute time. Throughput indicates the rate at which the sensor will receive data and which the system must keep up with. It may be specified by the input data cube size and the time required for the antenna to collect it. Generally an application will be broken into pipelined stages to increase the throughput capabilities; however, the latency requirement forces a hard time limit to the processing, causing limits to be decided upon and imposed for each stage. This trade-off is a major system design consideration.

¹ A CPI is the amount of time the radar antenna requires to send and receive all pulse information for a given data cube.

1.3 MORPHING MOTIVATION

The purpose of morphing is to increase the system capabilities in a significant way. In this document, we consider that three basic types of morph changes may take place: data dependent load balancing; changing from streaming to threaded computation; and changing parameter sets. These three types of morph changes were not chosen to necessarily fit within the taxonomies of morph types as shown in Figure 2 of [1]. These morph changes are considered under the control of the application (morph types 2 and 3) for the purposes of this document, but the same mechanisms used for the application API should be suitable for system and compiler level morphing as well. This section discusses the motivating reasons for the three morph changes discussed in this document.

The problem of data dependent load balancing is seen in GMTI through the need to perform operations on detected targets. This particular morph change is relevant for any parallel run of GMTI which might be performed. The problem occurs because computation will be parallelized across known axes of the data cube prior to the completion of CFAR (first portion of Detection stage). After CFAR, target detections will be spread across computing elements in an unknown way. In many systems, the radar will drop targets it is unable to process within its real-time limit. This leaves the choice between excess redundancy for CFAR parallelization, manually formulating a load balancing scheme, or accepting a larger target drop rate. The morph mechanism has the potential to provide an elegant solution to this problem by determining and performing appropriate resource allocation at run-time. This would most likely be considered as morph type 2 [1]. Proposals for how to indicate to the system that a morph type 0 [1] would be acceptable in one portion of the code, but not previous portions, would also be useful.

The change between streaming and threaded processing can cause problems due to the paradigm differences between the way we deal with each. While this boundary would generally be thought of as laying between the radar and the tracker, there is a boundary within GMTI which may be examined. The area in question is immediately after CFAR within the Detection stage. The CFAR detection operation has an input data cube of known size $((N_{rg}+N_{pc}-1) \times N_{cnb} \times N_{dop})$ as from Table 2) but outputs a target list with a length dependent upon the content of the input data. For this reason, CFAR would be processed in a “streaming” way and three-dimensional grouping and Estimation would then be performed as “threaded” operations. This morph makes assumptions causing this morph to be of type 3a [1]. Once again, system or compiler directives to make this a morph type of 1a or 4a [1] could also be investigated, and a combination of this morph change with the data dependent load balancing example could be used to investigate morph types 3b and 1b [1].

The final morph change for GMTI is both the most basic and the most complicated, the change of system parameters. A radar may have many missions to perform. The change may be very obvious as in a change between surveillance and target tracking. There may also be more subtle changes desirable to adjust the area and resolution of the radar scan. A PCA system must be able to adapt to the changing

resource demands associated with these changes. This morph change was designed to ideally be of morph type 3a, though implementations may end up using morph type 3b [1]. Once again, this same scenario could be used to investigate appropriate system and compiler directives for intelligent morphing of types 1a, 1b, or 4a [1].

2. MORPH CHANGE A: TARGET NUMBER AND DISTRIBUTION CHANGE

While the dynamic load balancing problem occurs in any detection system, an example stressful scenario would be airborne surveillance of a large, active area (such as a battlefield) where targets would be expected to converge, disperse, and move in all manner of patterns. Consider that the full ground extent to observe might occupy more than one scan in area, so separate data cubes would be collected for each area. Some scans would contain few targets, some would have many. Some scans may contain all targets grouped in one area, others may have multiple dense groups, and yet others may have widely dispersed targets. A simplified idea of what this might look like is illustrated in Figure 3 below.

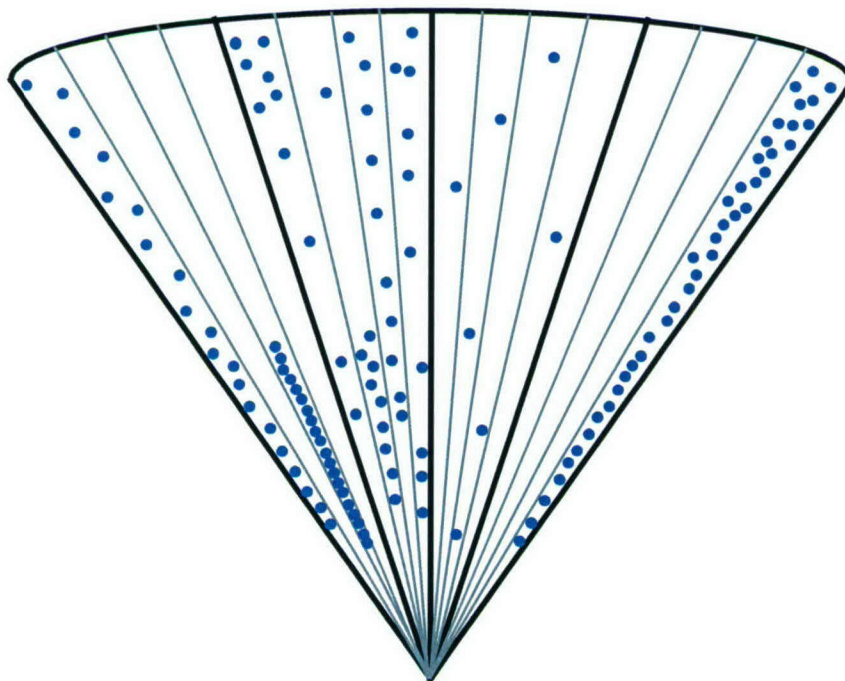


Figure 3. Target number and distribution morph change sample view. Bold lines indicate separate data cube areas. Thin lines indicate data slices for CFAR. Dots indicate targets.

To describe a quantitative example, we consider the execution of the Detection and Estimation stages, and constrain the mapping of the stages as follows. CFAR (first half of stage 6) is performed by all processing elements, and is assumed to take time C . Three dimensional grouping and Estimation (second half of stage 6 and stage 7) are assumed to take time P for a single processing element to estimate a single

target (assume for ease of analysis that each real target has an equal number of false targets detected with it to be eliminated by grouping). The overall latency requirement for this portion of the application is $C + 25P$; that is, there is enough time to detect all targets and to have a single processing element fully process 25 of them. We process four data cubes corresponding to four different areas and parallelize CFAR processing by azimuth for each cube. The distribution of targets in azimuth in each cube is shown in Table 3.

TABLE 3
Target Distribution Example

	Azimuth -1 to -1/2	Azimuth -1/2 to 0	Azimuth 0 to 1/2	Azimuth 1/2 to 1
Data cube 1	25	25	25	25
Data cube 2	33	33	33	1
Data cube 3	49	49	1	1
Data cube 4	97	1	1	1

A system at this point may choose between two options. The first option is to consider that each processing element can decide on its own which targets are most important and which must be dropped. The second option is to perform a load balancing operation to allow more targets to be detected. There are two measures of success which may then be used. The first measure is the percentage of targets detected. The second measure is the cost in processing elements used. For example, consider four example systems as shown in Table 4. These systems would score as shown in Table 5.

TABLE 4
Example System Descriptions for Load Balancing

1 Proc no bal	One processing element is used, no load balancing is attempted
4 Proc no bal	Four processing elements are used, no load balancing is attempted
16 Proc quad bal	Sixteen processing elements are used as four tightly coupled quads. Load balancing is performed within the quad but not externally (or alternately the data to the next level of division is evenly distributed)
4 Proc global bal	Four processing element are used, negligible latency load balancing is performed

TABLE 5
Success Metrics for Example Systems

	1 Proc no bal	4 Proc no bal	16 Proc quad bal	4 Proc global bal
Data cube 1 detection percentage	25%	100%	100%	100%
Data cube 2 detection percentage	25%	76%	100%	100%
Data cube 3 detection percentage	25%	52%	100%	100%
Data cube 4 detection percentage	25%	28%	100%	100%
Processor Cost	1	4	16	4

3. MORPH CHANGE B: STREAM- TO THREAD-BASED COMPUTATION

Before discussion on morphing between stream- and thread-based computation may take place, a few words need to be said by what exactly these terms will be defined to mean. Computation in general takes input data and produces output data. The concept of a streaming computation is one which may accept simple numerical inputs (scalars to N-dimensional arrays) which may be continuously fed in to the machine and have processed data continuously flow out. The concept of threaded computation is where a potentially more complicated set of data (a structure containing potentially different data types) will be passed as input, and output is also passed in one discrete portion when the entire operation is finished.

Using these definitions, the boundary between streaming and threaded computation falls within the Detection stage (after CFAR and before grouping). To demonstrate a stream-to-thread morph change, we use a scenario requiring 118 Mflop per data cube, with a CPI of 3.4 milliseconds, a latency requirement of 15 milliseconds, and a computational balance as shown in Table 6.

TABLE 6
Stream to Thread Example Computational Balance

Stage		Percentage of total flops
1	Time Delay & Equalization	25.121%
2	Adaptive Beamforming	20.640%
3	Pulse Compression	18.884%
4	Doppler Filtering	6.469%
5a	STAP weight computation	14.930%
5b	STAP weight application	7.058%
6	Detection	0.257%
7	Estimation	6.640%

Mapping this processing onto four, balanced, pipelined stages would have the first stage performing Time Delay & Equalization (1); the second stage would perform Adaptive Beamforming (2); the third would perform Pulse Compression and Doppler Filtering (3 & 4); and the final would contain STAP weight computation, STAP weight application, Detection, and Estimation (5–7). The percentage of time taken by STAP weight application, Detection, and Estimation is slightly less than that for STAP weight computation (13.955% vs. 14.930%), so a logical mapping of the fourth stage would be to give half of this stage's resources to weight computation (Stage 5a) and the other half to the other kernels (Stages 5b–7). To

avoid incurring excess communication costs, the mapping to show this morph change would allocate resources to STAP weight application, Detection, and Estimation based upon time as opposed to space (see Figure 4). The top row of boxes represents the data processed by the first half of the resources (Resource Set A), which always performs STAP weight computation (5a). The bottom row of boxes represents the data processed by the second half of the resources (Resource Set B), which will alternate between performing STAP weight application (5b) and Detection and Estimation (6 & 7).

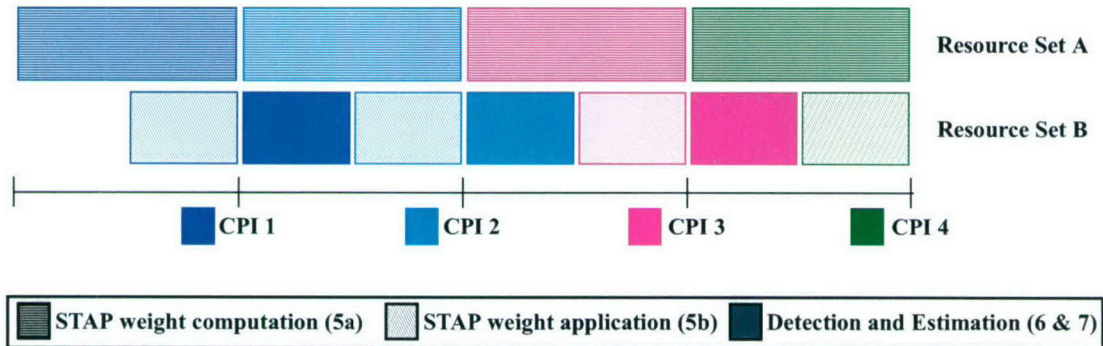


Figure 4. Time-shared resource mapping for STAP, Detection, and Estimation.

To state this in words, STAP weight computation for the current CPI will be performed on Resource Set A at all times. During the first half of the CPI, the Detection and Estimation processing for the previous CPI will be performed on Resource Set B (which will roughly use up the time past four CPIs in the latency requirement). It will then switch to perform STAP weight application for the current CPI during the second half of the CPI. This will have Resource Set B performing five types of computations, two streaming (matrix multiplication and CFAR) and three threaded (three-dimensional grouping, spline interpolation, and Maximum Likelihood Estimation). The primary point is to investigate if there are any significant costs for the target architecture to execute a mix of streaming and threaded kernels, compared to executing optimal configurations for two kernels of the same type. As no actual change in morph state takes place in Resource Set A, there is no need to consider its implementation on PCA hardware to investigate this morph change.

4. MORPH CHANGE C: PARAMETER SET CHANGE

Parameter changes can be useful in most systems. As a convenient example we consider a space-based radar (SBR). It would be desirable for such a system to have modes for examining different types of ground situations. Table 7 below gives relevant parameters for a few example GMTI morph states and Table 8 gives system performance information.

TABLE 7
GMTI Morph State Parameters

Parameter	Default	Wilderness	City
PRF	2,000	1,500	3,000
Fsampling	5,000,000	3,500,000	5,000,000
Nrg	2,250	2,100	1,500
Npri	17	65	65
Npc	250	234	167
Ndop	16	64	64
MaxTargets	100	25	375
Nfam	10	10	20

TABLE 8
GMTI Morph State Requirements

	Default	Wilderness	City
Stream Flops (Mflop) (stages 1-6) ^a	210	540	462
Thread Flops (Mflop) (stage 7)	11	2	78
Total Flops	221	542	539
CPI time (milliseconds)	8.5	43	22
Computational Throughput (Gflop/sec)	26.0	12.5	24.9

^a Three-dimensional grouping performs only comparisons, so does not contribute to the flop count. For this reason the Detection stage's flops may be considered all streaming.

The concept of the Wilderness parameter set would be to facilitate the radar's examination of a large area on the ground, not requiring sharp resolution, and with the expectation of few targets. The City set would be for examining a small area on the ground, requiring sharp resolution, and would expect a large number of targets. Note that the numbers in the above tables represent a narrowband SBR system (which would generally be wideband, but the above would relate to one subband stream).

An interesting example morph change would be between the Wilderness and City sets. The first thing to note about these parameter sets is the Nfam parameter, which indicates antenna area. The Wilderness scenario, which uses half the antenna area compared to the City scenario, could conceivably receive two data cubes while the City scenario receives one. In this manner, these two scenarios are roughly balanced in computational throughput requirements. The balance between stage requirements has changed though (see Figure 5 below).

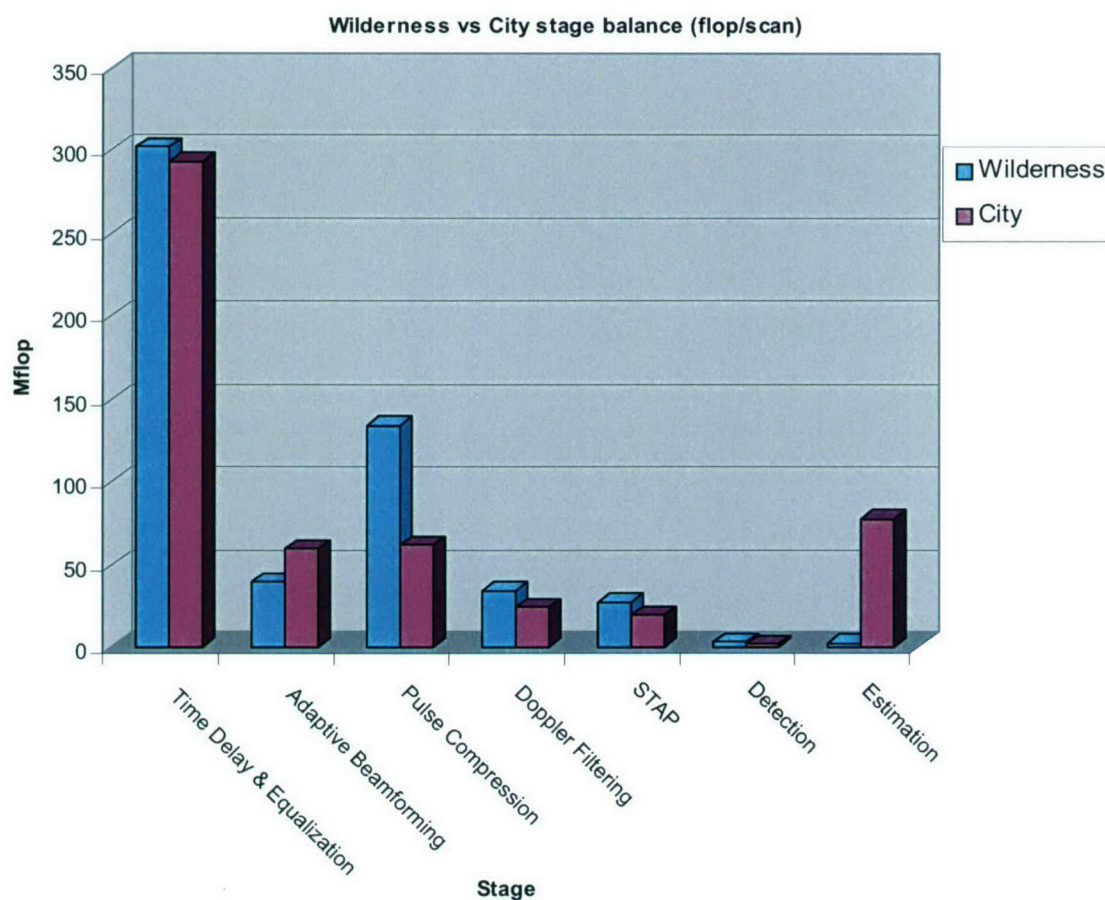


Figure 5. Flop comparison between Wilderness and City scenarios.

The most significant change between these two parameter sets is the drastic increase of the requirements for the Estimation stage, pushing the threaded computation up to nearly fifteen percent of the workload. Note also that the difference in the number of range gates will change either dimension or operation frequency for every stage of computation, altering the manner in which the computational stages may scale. The investigation of how to adjust the scales of each stage and the appropriate hardware reconfiguration is the challenge for this morph change.

5. CONCLUSION

In summary, we have described three significant areas where PCA might demonstrate its capacity for improvement for a general radar: dynamic load balancing; streaming and threaded computation optimally performed on the same hardware; and system parameter changes. We are providing these examples to the community with the hope of getting feedback on the ways that real system designers can make use of PCA hardware.

REFERENCES

- [1] Campbell, Daniel P.; Cotel, Dennis M.; Judd, Randall R.; and Richards, Mark A., "Introduction to Morphware: Software Architectures for Polymorphous Computing Architectures," version 0.4, Georgia Institute of Technology and Space and Naval Warfare Systems Center San Diego, December 2003.
- [2] Coate, William G., "Preliminary Design Review: Kinematic Tracking for the PCA Integrated Radar-Tracker Application," MIT Lincoln Laboratory Project Report PCA-IRT-4, 25 February 2003, issued 6 February 2004.
- [3] Reuther, Albert I., "Preliminary Design Review: GMTI Narrowband for the Basic PCA Integrated Radar-Tracker Application," MIT Lincoln Laboratory Project Report PCA-IRT-3, 27 February 2003, issued 6 February 2004.

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) 20 February 2004		2. REPORT TYPE Project Report		3. DATES COVERED (From - To)	
4. TITLE AND SUBTITLE Morphing Scenarios for the GMTI Portion of the PCA Integrated Radar Tracker				5a. CONTRACT NUMBER F19628-00-C-0002	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) W. G. Coate and J. M. Lebak				5d. PROJECT NUMBER 1084	
				5e. TASK NUMBER 0	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) MIT Lincoln Laboratory 244 Wood Street Lexington, MA 02420-9108				8. PERFORMING ORGANIZATION REPORT NUMBER PR-PCA-IRT-6	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) DARPA/IPTO 3701 Fairfax Drive Arlington, VA 22203-1714				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S) ESC-TR-2005-053	
12. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT The key to the success of the DARPA-sponsored Polymorphous Computing Architectures (PCA) program is the proper development of the morphing concept. One of MIT Lincoln Laboratory's contributions towards this effort is the Integrated Radar-Tracker (IRT) application. The initial IRT consists of a Ground Moving Target Indicator (GMTI) radar and a kinematic tracker (KT). As GMTI has received more community interest, it is used in this first document as a background for identification and discussion of promising computational issues which may be addressed by morphing. This document examines three areas from GMTI: data dependent load balancing; changing from streaming to threaded computation; and changing parameter sets.					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report	18. NUMBER OF PAGES 20	19a. NAME OF RESPONSIBLE PERSON
a. REPORT Unclassified	b. ABSTRACT Same as Report	c. THIS PAGE Same as Report			19b. TELEPHONE NUMBER (include area code)